Universidad Autónoma de Madrid
Escuela politécnica superior

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Trabajo Fin de Grado

DESARROLLO DE UN MARCO DE TRABAJO PARA SEGMENTACIÓN SEMÁNTICA EN BASES DE DATOS DE IMÁGENES URBANAS

Autor: Javier González Cabrero
Tutor: Pablo Carballeira López
Ponente: Jesús Bescós Cano

Junio 2020

DESARROLLO DE UN MARCO DE TRABAJO PARA SEGMENTACIÓN SEMÁNTICA EN BASES DE DATOS DE IMÁGENES URBANAS

Autor: Javier González Cabrero
Tutor: Pablo Carballeira López
Ponente: Jesús Bescós Cano

Video Processing and Understanding Lab
Dpto. de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio 2020

Trabajo parcialmente financiado por el Gobierno de España bajo el proyecto TEC2017-88169-R (MobiNetVideo)

Resumen

Resumen

La navegación a través de entornos no estructurados es una capacidad básica de las criaturas inteligentes, y por lo tanto, es de interés fundamental en el estudio y desarrollo de la inteligencia artificial. La capacidad para automatizarlo, pudiendo navegar sin el uso de mapas, sólo con imágenes a través de entornos urbanos está siendo objetivo de incipientes trabajos. Los sistemas de navegación automática se basan en sistemas entrenados con imágenes del entorno urbano, de la ciudad. La segregación de elementos fijos y móviles puede ser útil para conseguir mejorar el proceso de entrenamiento de estos sistemas, haciendo que su aprendizaje se base en la apariencia de elementos fijos, y no en elementos móviles que pueden distorsionar el proceso de aprendizaje, y por lo tanto, funcionamiento. En este contexto, la segmentación semántica de los objetos podría ayudar a la mejora del sistema de localización y guiado.

El objetivo del trabajo es desarrollar un marco de trabajo para segmentación semántica en bases de datos de imágenes urbanas. Aprovechando la disponibilidad de Google Street View, se utiliza como base de datos para la implementación del trabajo por su cobertura mundial y contenido fotográfico, mostrando imágenes de distintas localizaciones con distintas características de cámara como, por ejemplo, el campo de visión (fov), el ángulo de toma de la imagen, tanto horizontal (heading) como vertical (pitch), etc. Esto hace que de una sola localización se pueda abarcar los 360º con imágenes, mostrando diferentes puntos de vista de la localización. Como hay disponibilidad de información 360º, en este marco de trabajo se incluye la reproyección de máscaras semánticas en la esfera para agregar la información redundante de los diferentes puntos de vista. Ante esto, surge la siguiente hipótesis, ¿la utilización de los diferentes puntos de vista podría ayudar a mejorar la segmentación semántica que se obtiene de un solo punto de vista? Para evaluar y responder a esto se establece un marco de evaluación de diferentes algoritmos de reproyección y agregación. Los métodos de agregación que se proponen son sencillos.

Este trabajo recoge un conjunto de conclusiones preliminares sobre un conjunto limitado de datos, mostrando que en la mayoría de casos los resultados obtenidos por los métodos de agregación sencillos implementados no superan los obtenidos por segmentación directa.

Palabras Clave

Navegación automática, imágenes urbanas,heading, pitch, field of view, agregación, segmentación semántica, reproyección, dataset.

Abstract

Navigation through unstructured environments is a basic ability of intelligent creatures, and therefore, is of fundamental interest in the study and development of artificial intelligence. The capacity to automate it, being able to navigate without the use of maps, only with images through urban environments is being the target of incipient works. The automatic navigation systems are based on systems trained with images of the urban environment, of the city. The segregation of fixed and mobile elements can be useful to improve the training process of these systems, making their learning based on the appearance of fixed elements, and not on mobile elements that can distort the learning process, and therefore, operation. In this context, the semantic segmentation of objects could help to improve the location and guidance system.

The aim of the work is to develop a framework for semantic segmentation in urban image databases. Taking advantage of the availability of Google Street View, it is used as a database for the implementation of the work because of its worldwide coverage and photographic content, showing images of different locations with different camera characteristics such as field of view, heading, pitch, etc. This makes it possible to cover the 360º with images from a single location, showing different points of view of the location. As 360º information is available, this framework includes the reprojection of semantic masks in the sphere to add the redundant information of the different points of view. In view of this, the following hypothesis arises: could the use of the different points of view help to improve the semantic segmentation obtained from a single point of view? To assess and respond to this, a framework for evaluating different reprojection and aggregation algorithms is established. The proposed aggregation methods are simple.

This work gathers a set of preliminary conclusions on a limited set of data, showing that in most cases the results obtained by the simple aggregation methods implemented do not exceed those obtained by direct segmentation.

Key words

Automatic navigation, city images, heading, pitch, field of view, aggregation, semantic segmentation, reprojection, dataset.

Agradecimientos

En primer lugar, quiero dar las gracias a mi tutor, Pablo, por su ayuda, su paciencia y su enseñanza continua durante este trabajo.

Agradecer especialmente a mi familia, que ha estado apoyándome y, sobre todo, aguantándome siempre, y en especial, en estos momentos de pandemia.

Finalmente, a esos que un día fueron compañeros de clase y a día de hoy son grandes amigos.

Índice general

Índice de Figuras

Índice de Tablas

1 Introducción

A continuación se introducen las motivaciones y el contexto en el que se desarrolla este trabajo. Además, se presentan los objetivos y subtareas del proyecto. Para finalizar, se hace un resumen de la estructura de la memoria.

1.1. Motivación

Hoy en día uno de los tipos más poderosos de inteligencia artificial es la visión artificial, que está presente en el día a día de las personas sin que se den cuenta. La visión artificial está en constante desarrollo y crece a pasos agigantados, se prevé que para 2022 el mercado de la visión artificial alcance los 48.600 millones de dólares [11]. Esto es realmente relevante, ya que hasta hace poco sólo funcionaba con una capacidad limitada, pero gracias a los avances en la inteligencia artificial y a las innovaciones en el aprendizaje profundo y las redes neuronales, el campo de visión artificial ha podido dar grandes saltos en los últimos años y superar a los humanos en algunas tareas relacionadas con la detección y el etiquetado de objetos. Uno de los factores que impulsan este crecimiento es la cantidad de datos que se generan hoy en día y que luego se utilizan para entrenar y mejorar la visión artificial.

Hay múltiples aplicaciones de la visión artificial como en la seguridad, en la conducción autónoma, en la agricultura, en la industria, en la banca, en la atención médica entre muchas otras. La navegación automática es otra de ellas, basada en la navegación sin el uso de mapas [12], sólo con imágenes de las localizaciones urbanas y en la que se va a centrar este trabajo. En el desarrollo de un algoritmo de visión artificial para el guiado por una ciudad, se puede servir de la información visual de una cámara, que se lleva encima, para su localización y guiado. De esta información visual puede ser más útil la apariencia de los objetos estáticos, que pueden ayudar a la localización, que la de los objetos móviles, que al ser temporales son menos determinantes y menos fiables. Por tanto, sería útil introducir esta división en objetos fijos y móviles a los algoritmos existentes. Además, si el algoritmo de guiado incluye un sistema de segmentación semántica de los diferentes objetos podría ayudar a la mejora del sistema de localización y guiado.

La cantidad de datos generados día a día permite desarrollar bases de datos para cada aplicación, permitiendo su entrenamiento y por tanto, su mejora. Las bases de datos de Google

Street View son una de las más usadas por su variedad y cantidad de imágenes urbanas, estas bases de datos pueden a ayudar a entrenar un modelo de aprendizaje automático para hacer la localización y el guiado automático. En un sola localización se puede abarcar los 360º con imágenes, mostrando diferentes puntos de vista, por lo que surge la cuestión de qué ocurriría si se utilizan los diferentes puntos de vista para realizar la segmentación semántica. Por tanto, para responder a esta cuestión se desarrolla un marco de trabajo para segmentación semántica en bases de datos de imágenes urbanas.

La motivación principal es hacer una evaluación inicial, con el fin de analizar si con el uso de múltiples puntos de vista de una localización mejorará la segmentación semántica que se puede obtener de un solo punto de vista.

1.2. Objetivos

Los dos objetivos principales de este trabajo son: (1) La creación de un marco de trabajo para la reproyección de las máscaras semánticas de 360º, con varios puntos de vista, de una localización en una sola máscara semántica y (2) la comparación y evaluación de los resultados obtenidos de una misma localización en el marco de trabajo anterior con respecto a una imagen de un sólo punto de vista obtenida del segmentador semántico. Para poder realizar los objetivos principales se plantean las siguientes subtareas:

- Estudio de las bases de datos de Google Street View, sus APIs y las herramientas para la extracción de imágenes. Análisis de distintos modelos y datasets para la segmentación semántica y herramientas donde desarrollarlos. Observación de los sistemas del estado del arte sobre reproyección.
- Desarrollo del marco de trabajo, incluyendo herramientas del punto anterior, automatizando la conversión de unas a otras e implementando un esquema de reproyección de imágenes de Google Street View y agregación de la información redundante de dichas imágenes.
- Evaluación final y comparación de resultados entre los diferentes métodos de agregación propuestas y un segmentador semántico ó segmentación directa.
- Evaluación de la influencia de los parámetros de las imágenes de Google Street View en los resultados

1.3. Documentación

La estructura del trabajo está organizada de la siguiente manera: En el Capítulo 2, se analiza el estado del arte para la obtención y extracción de imágenes de Google Street View, también se analiza la segmentación semántica, su funcionamiento y sus distintas aplicaciones. Por último, se estudia el funcionamiento de una herramienta reproyección. En el Capítulo 3, se encuentra el esquema del sistema en la Sección 3.2, que permite entender el funcionamiento del sistema desarrollado y las herramientas utilizadas en el mismo, que son desarrollados en las siguientes secciones. En el Capítulo 4, primero se observan las condiciones de los experimentos, las bases de datos y las métricas que se van a utilizar para medir los resultados. Y ya en la Sección 4.4, se exponen los resultados obtenidos. Por último, en el Capítulo 5 se podrán ver las conclusiones sobre el trabajo y posibles trabajos futuros.

2

Estado del arte

2.1. Introducción

Durante los últimos años, la segmentación semántica ha sido uno de los problemas más estudiados en el campo de la visión artificial por su comprensión a nivel del píxel de las imágenes siendo realmente útil para la conducción autónoma, imágenes médicas o navegación automática entre muchas otras. Este trabajo se enfoca en la navegación automática, por tanto este capítulo tendrá como objetivo mostrar que imágenes se usan, cómo extraer dichas imágenes y generar a partir de ellas una base de datos, además de profundizar en el estudio de la segmentación semántica conociendo cómo funciona, cuáles son algunos de sus datasets y arquitecturas más utilizados. Por último, se estudiará una de las herramientas utilizadas para las reproyección de imágenes.

2.2. Imágenes de Google Street View

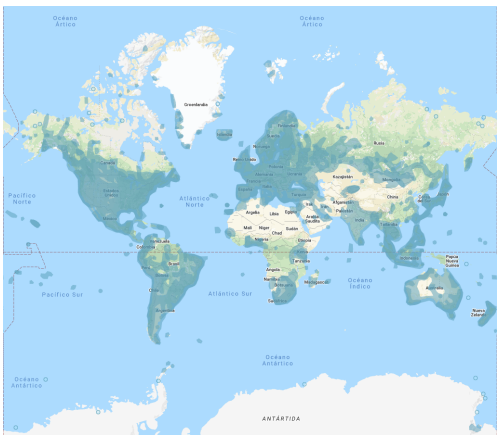


Figura 2.1: Radio de acción de Google Street View. En azul se puede observas las zonas donde ha llegado Google Street View. Extraído de [1]

La navegación automática necesita de imágenes urbanas, y por ello, se utiliza Google Street View [1] que es una representación virtual del entorno que nos rodea en Google Maps, que contiene millones de imágenes panorámicas. Todas estas imágenes son obtenidas por Google o por sus colaboradores. Street View fue lanzado 2007 en varias ciudades de Estados Unidos, y desde entonces se ha ampliado para incluir ciudades y zonas rurales de todo el mundo como se puede observar en la Figura 2.1 llegando a capturar más de 16 millones de kilómetros.

Figura 2.2: Imagen stitched. Extraído de [2]

Para la toma de estas imágenes se utilizan cámaras 360º, proporcionando 360º de movimiento horizontal y 180º de movimiento vertical. Por lo que hay múltiples puntos de vista de la misma posición cubriendo los 360º, mediante la superposición de estas vistas se compone una imagen panorámica. Por tanto, Google Street View muestra panorámicas de imágenes stitched [13], es decir, combinación de imágenes para producir una imagen panorámica usando un software específico como se puede observar en la Figura 2.2.

2.2.1. Street View Static API

Street View Static API [14] es una biblioteca de funciones, procedimientos y subrutinas que permite extraer imágenes de la base de datos de Google Street View para su uso y posterior manipulación con el fin de ayudar a desarrollar el propósito de este trabajo. La solicitud de una imagen extrae una subimagen no panorámica de las que está formada la imagen panorámica de 360º. Esta subimagen se especifica mediante unos parámetros que se describen a continuación:

Parámetros obligatorios:

- **Location/Pano**: Puede ser una cadena de texto (como Chagrin Falls, OH) o un valor latitud/longitud (40.457375,-80.009353). Street View Static API se ajustará a la imagen panorámica más cercana a esta ubicación. Cuando se proporciona una latitud/longitud, la API busca en un radio de 50 metros la fotografía más cercana a esta ubicación. Dado que las imágenes de Street View se actualizan periódicamente y las fotografías pueden tomarse desde posiciones ligeramente diferentes cada vez, es posible que su ubicación se ajuste a una imagen panorámica diferente cuando se actualicen las imágenes.

- **Size**: Especifica el tamaño de salida de la imagen en píxeles. El tamaño se especifica como anchoxaltura, por ejemplo, tamaño=600x400 devuelve una imagen de 600 píxeles de ancho, y 400 de alto.

Parámetros opcionales:

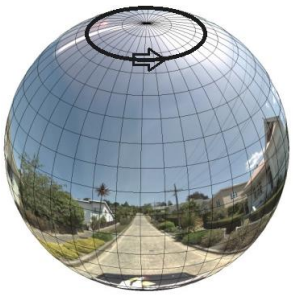


Figura 2.3: Variación de heading de una imagen sobre proyección en una esfera. Extraído de [3]

(a) Heading = 0º. (b) Heading = 20º. (c) Heading = 40º.

Figura 2.4: Ejemplo de tres imágenes de la misma ubicación con diferente heading.

- **Heading**: (Valor por defecto es 90). Indica la dirección de la brújula de la cámara. Los valores aceptados son de 0º a 360º (ambos valores indican el Norte, con 90º indicando el Este, 180º el Sur, y 270º el Oeste). Los variación de los valores es sobre el eje horizontal de la esfera como se muestra en la Figura 2.3 . Ejemplo gráfico de la variación de heading se puede observar en la Figura 2.4.

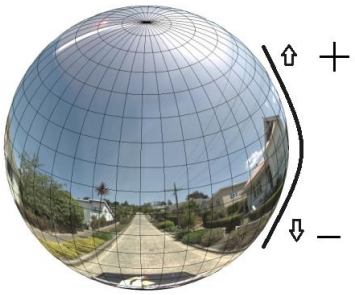


Figura 2.5: Variación de pitch de una imagen sobre proyección en una esfera. Extraído de [3]

(a) Pitch = -20º. (b) Pitch = 0º. (c) Pitch = 20º.

Figura 2.6: Ejemplo de tres imágenes de la misma ubicación con diferente pitch.

- **Pitch**: (Valor por defecto es 0). Especifica el ángulo de inclinación de la cámara en relación con el plano del suelo. Los valores positivos inclinan la cámara hacia arriba y los valores negativos inclinan la cámara hacia abajo. Los variación de los valores es sobre el eje vertical de la esfera como se muestra en la Figura 2.5 y unos valores aceptados desde -90º a +90º. Ejemplo gráfico de la variación de heading se puede observar en la Figura 2.6.

(a) FOV = 20º. (b) FOV = 50º.

(c) FOV = 90º. (d) FOV = 120º.

Figura 2.7: Ejemplo de tres imágenes de la misma ubicación con diferente FoV.

- **FOV(Field of View)**: (Valor por defecto es 90). Determina el campo de visión horizontal de la imagen. El campo de visión se expresa en grados, con un valor máximo permitido de 120º. Valores más pequeños de FOV, indican un mayor nivel de zoom. Ejemplo gráfico de la variación de heading se puede observar en la Figura 2.7.

- **Radius**:(Valor por defecto es 50) Establece un radio, especificado en metros, en el que buscar una imagen panorámica, centrado en la latitud y longitud dadas. Los valores válidos de entrada son enteros no negativos.

2.3. Herramienta Street View Extractor

Con el fin de automatizar la extracción de imágenes de Google Street View utilizamos la herramienta StreetView-Extractor [15] que descarga imágenes a lo largo de la ruta especificada, o imágenes de una dirección concreta, denominado como modo single y que se utiliza en el desarrollo de este trabajo. Esta herramienta, y el modo single en concreto, tiene gran utilidad en este trabajo ya que dada una localización permite extraer todas las imágenes cubriendo la esfera 360º con dos parámetros que son dos ángulos, el heading y el pitch. En ambos ángulos se puede definir un valor inicial, final y una tasa de muestreo entre ambos valores, automatizando la extracción de imágenes para estos valores especificados y generando bases de datos de imágenes urbanas de densidad variable en función de los valores de los ángulos descritos anteriormente. Esta herramienta está basada en la API de Street View, explicada en la Sección 2.2.1, y en la API de Directions, que extrae muestreos de rutas para obtener las imágenes de las coordenadas determinadas. Esta herramienta se desarrolla en el Trabajo de Fin de Grado de Paula Guerra [15], junto con otros miembros del VPULab. A su vez, la herramienta está basada en el repositorio original de GitHub [16]. En el Trabajo de Paula, se llevan a cabo unas modificaciones con

respecto a la versión original, permitiendo variar parámetros que originalmente eran fijos, como el heading, pitch y FoV, posibilitando que la herramienta haga un barrido de parámetros lo que es muy útil para este contexto. Entre todos los parámetros que la herramienta ofrece, en este trabajo, se utilizan los parámetros de heading, pitch, fov y el tamaño de las imágenes explicados anteriormente, y el parámetro de origen, que se trata de la localización de la cual se extraen las imágenes.

2.4. Segmentación semántica

2.4.1. Introducción

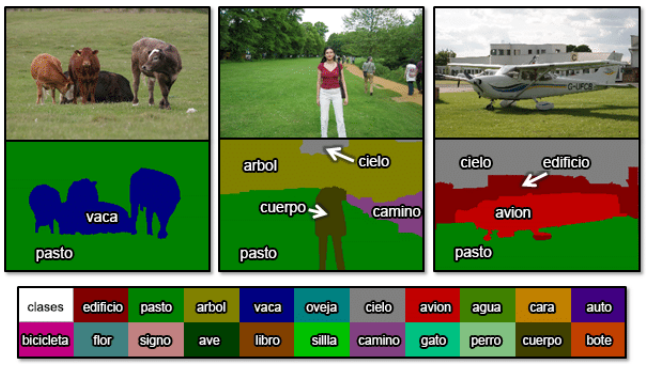


Figura 2.8: Ejemplo de segmentación semántica. Extraído de [4]

La segmentación semántica [17] [18] se enmarca en el contexto de las tareas clásicas de visión artificial, y se trata del proceso de asignar una etiqueta a cada píxel de la imagen. Estas clases podrían ser coche, árbol, vaca, camino... como se puede observar en la Figura 2.8

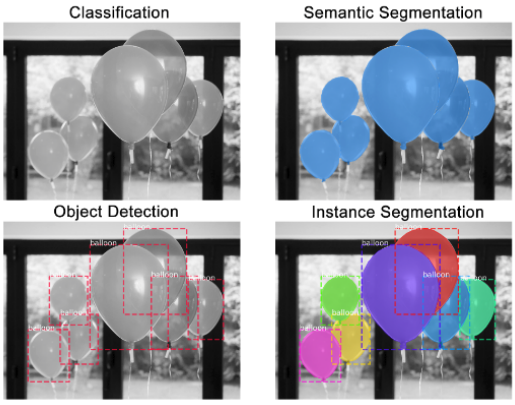


Figura 2.9: Diferencia entre el reconocimiento de imágenes, detección de objetos, segmentación semántica y segmentación de instancias. Extraído de [5]

Es diferente al reconocimiento de imágenes [19] que asigna una etiqueta a toda la imagen, también diferente a la detección de objetos [20], que localiza los objetos encajándolos en un bounding box. La segmentación semántica trata los múltiples objetos de la misma como una sola entidad, lo que hace que sea diferente a la segmentación de instancias [21], que trata los múltiples objetos de la misma clase como objetos individuales distintos, este tipo de segmentación puede ser muy

útil en aplicaciones que se utilizan para contar el número de objetos. Haciendo esta segmentación más compleja que la segmentación semántica. Ejemplo de estas diferencias [22] se puede observar en la Figura 2.9

La segmentación semántica permite descomponer una imagen en partes significativas y entender la escena a un nivel más granular. Mientras que la clasificación de la imagen es útil para ver lo que hay en la imagen desde un plano general. La detección de objetos permite localizar el contenido de una imagen y la segmentación permite definir las formas y los límites de los objetos de la imagen.

2.4.2. Aplicaciones

La capacidad de la segmentación semántica de poder definir las formas y los límites de los objetos con precisión de la imagen resulta muy útil para aplicaciones en diversos campos como los siguientes [23]:

- **Conducción autónoma**: La necesidad de interpretar y reaccionar en tiempo real hace que el sistema de cámaras de un vehículo autónomo necesite crear un mapa de lo que ve a nivel de píxel para poder navegarlo de forma segura y eficiente identificando correctamente otros vehículos, peatones, aceras, semáforos...
- **Inspección industrial**: Detección de imperfecciones en materiales y equipos fabricados.
- **Análisis de imágenes por satélite**: Identificación de elementos del terreno.
- **Análisis de imágenes médicas**: Técnica poderosa en los procesos de diagnóstico y tratamiento que requieran imágenes médicas.
- **Robótica**: Para realizar de manera eficiente tareas industriales, de servicios, agrícolas, etc.
- **Herramientas creativas**: Muchas utilidades para la edición de imágenes y vídeos.

2.4.3. ¿Cómo funciona?

En la actualidad, como en muchas otras tareas de visión artificial, los algoritmos se basan en redes neuronales. Las más usadas para la segmentación semántica son convolucionales(CNN). Una CNN [24] es un algoritmo para Deep Learning, que es un tipo de machine learning en el que un modelo aprende a realizar diferentes tareas a partir de imágenes, vídeos, textos o sonidos.
Este trabajo se va a centrar en las que aplican a imágenes. Una CNN puede tener cientos de capas que aprendan a detectar distintas características de la imagen de entrada. Se aplican filtros a la imagen de entrada de entrenamiento y la salida convolucionada de la imagen se emplea como entrada para la siguiente etapa. Estos filtros convolucionales son bidimensionales y se aplican a las matrices de la imagen de entrada con parámetros que se modifican en entrenamiento para que la red aprenda una tarea. Estos filtros son la base de las CNNs y el proceso distintivo de las mismas, que por ejemplo, lo diferencia de la redes neuronales clásicas, donde todas las neuronas están interconectadas entre sí. Aunque las redes neuronales utilizadas para la segmentación de imágenes pueden diferir ligeramente en su aplicación, la mayoría tienen una estructura básica similar a la de una CNN, que se trata de una arquitectura de codificador-decodificador. El codificador es un conjunto de capas que extraen características de una imagen a través de una secuencia de filtros progresivamente más estrechos y profundos. La reducción de la dimensión espacial(submuestreo) de la imagen de una capa a otra se realiza mediante una capa pooling(en

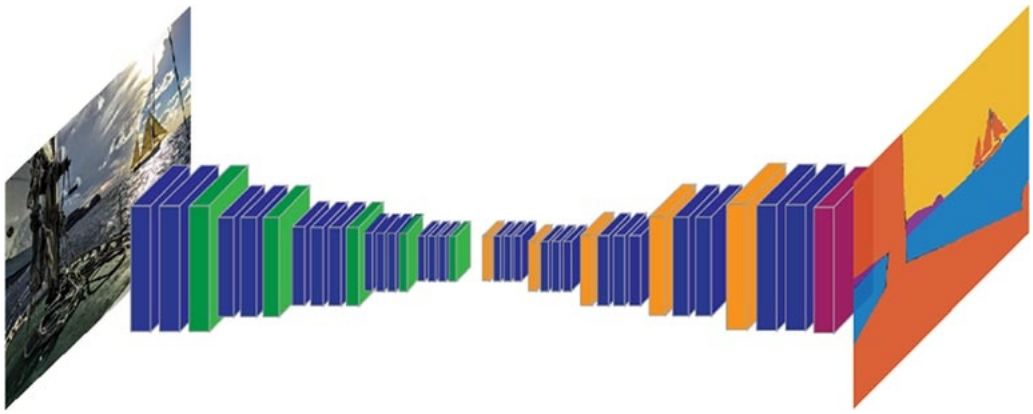


Figura 2.10: Ejemplo de una CNN realizando una segmentación semántica. Extraído de [6]

color verde en la Figura 2.10). El decodificador es un conjunto de capas que aumenta progresivamente la salida del codificador para asegurar que el tamaño final es igual que el de entrada, el decodificador aumenta la dimensión espacial(sobremuestreo) de la imagen el mismo número de veces que se hizo el submuestreo. Y al igual que en el codificador, el sobremuestreo de una capa a otra se realiza mediante una capa unpooling(en color naranja en la Figura 2.10). Al final del decodificador encontramos una capa, que hace funciones de máscara semántica, asignando cada píxel a una clase [6]. Todo esto permite la segmentación semántica.

Para poder entrenar la red con el fin de realizar la segmentación semántica se necesita una base de datos con imágenes ya segmentadas, a continuación se exponen brevemente las más populares:

- **Datasets de imágenes de propósito general**:
  - **Common Objects in COntext(COCO) [25]**: Con más de 200k imágenes etiquetadas, 1,5 millones de objetos instanciados, y 182 categorías de objetos COCO es un conjunto de datos desarrollado por Microsoft para segmentación, detección y clasificación de objetos. Esas categorías se podrían dividir en dos, por un lado objetos con una forma bien definida, por ejemplo, coche o persona que se denomina COCO-Thing. Y por otro, aquellas cosas con formas amorfas o regiones de fondo, por ejemplo, la hierba o el cielo que se denomina COCO-Stuff. Ambas distribuciones tienen el mismo número de categorías, 91. La jerarquía de ambos se puede observar en la Figura 2.11.
  - **PASCAL Visual Object Classes (PASCAL VOC) [26]**: Uno de los datasets más usados para segmentación, detección y clasificación de objetos, tiene alrededor de 10000 imágenes anotadas semánticamente con 21 clases diferentes.
- **Datasets de imágenes urbanas**:
  - **Cityscapes [27]**: Con alrededor de 3000 imágenes clasificadas en 30 clases. Estas clases se pueden encuadrar en 8 grupos. Las imágenes de este dataset son urbanas tomadas por carretera.
  - **Otros**: Hay otros datasets alternativos como CamVid [28], basada en imágenes en movimiento que contiene 32 clases semánticas, KITTI [29] y SYNTHIA [30].

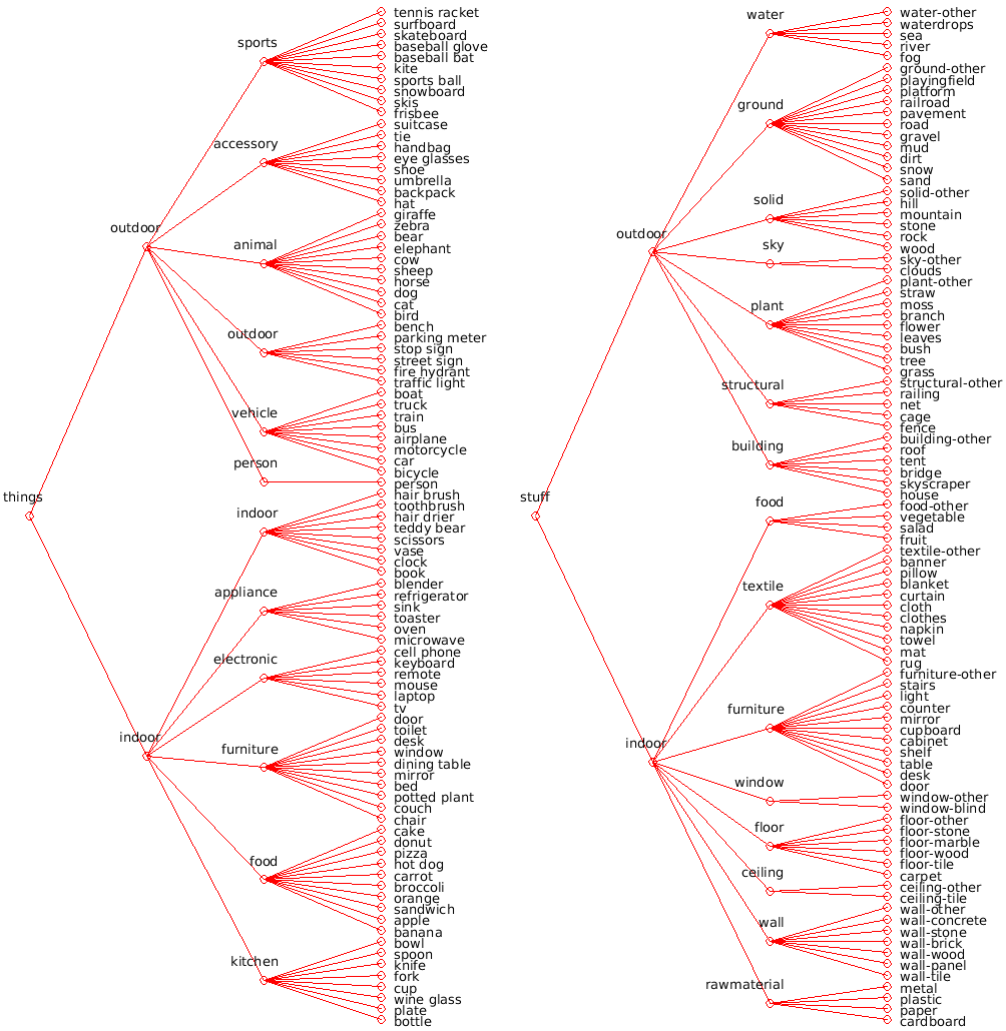


Figura 2.11: Estrcutura de clases del dataset de COCO-Stuff y COCO-Thing. Extraído de [7]

Los métodos del estado del arte utilizados para la segmentación semántica se podrían dividir en dos categorías [31]:

- **Region-based semantic segmentation [32]**: Segmentación mediante reconocimiento. Primero extrae las regiones de una imagen y las describe para después realizar una clasificación de las regiones. Las predicciones basadas en regiones se transforman en predicciones de píxeles, generalmente etiquetando un píxel según la región de mayor puntuación que lo contiene. Arquitecturas típicas usadas en estos método: SPP(Spatial Pyramidal Pooling) [33], Faster R-CNN [34], Mask R-CNN [35] o MPA(Multi-scale Patch Aggregation) [36].
- **FCN-based semantic segmentation [37]**: Una red neural totalmente convolutiva (FCN) es una CNN normal, en la que la última capa totalmente conectada es sustituida por otra capa convolutiva con un gran ”campo receptivo”. Por tanto, la diferencia principal con la familia anterior es que no tienen capas totalmente conectadas. Arquitecturas típicas usadas en estos método: ParseNet [38], PSPNet(Pyramidal Scene Parsing) [39], FPN(Feature Pyramid Network) [40], U-Net [41] o Deeplab, que explicará con detalle más adelante.

En este trabajo se ha usado DeepLab[42], que es un modelo lanzado por Google en 2016 inspirado en FCN. Ha tenido varias versiones desde su primer lanzamiento [43]:

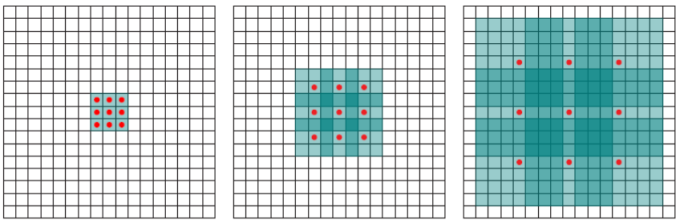


Figura 2.12: Funcionamiento de la convolución dilatada: En la subiamgen a, el filtro se produce por una convolución de una dilatación, en el que cada elemento en el filtro tiene un campo receptivo de 3x3. En la subimagen b, el filtro nuevo se produce a partir del filtro anterior por una convolución de dos dilataciones; en el que cada elemento en en el filtro nuevo tiene un campo receptivo de 7x7. En la subiamgen c, el filtro nuevo se produce a partir del filtro de la subimagen b por una convolución de cuatro dilataciones en el que cada elemento en filtro nuevo tiene un campo receptivo de 15x15. El número de parámetros asociados a cada capa es idéntico. El campo receptivo crece exponencialmente mientras que el número de parámetros crece linealmente. Extraído de [8]

- **DeepLab-v1 [44]**: Se usa una convolución atrous(atómica) para controlar la resolución a la que se calculan las respuestas de las características dentro de las CNNs. Esto también se conoce como convolución dilatada e introduce otro parámetro, la tasa de dilatación, que espacia los píxeles en un campo de visión más amplio. La dilatación sistemática permite la expansión exponencial del campo receptivo sin pérdida de resolución o cobertura. Un ejemplo gráfico se puede observar en la Figura 2.12.
- **DeepLab-v2 [45]**: Utiliza ASPP(Atrous Spatial Pyramid Pooling) para segmentar robustamente los objetos a múltiples escalas con filtros a diferentes velocidades de muestreo y campos de visión efectivos, mejorando la precisión. Tanto la versión 1 como la 2 usa la herramienta de post-procesado denominada CRF(Conditional Random Field) que mejora la segmentación.
- **DeepLab-v3** [46]: Se añaden al ASPP funciones a nivel de imagen y se aplica normalización por grupos para facilitar el entrenamiento.
- **DeepLab-v3+ [47]**: Se añade a la versión anterior un módulo decodificador simple pero efectivo para perfeccionar los resultados de la segmentación, especialmente en los límites de los objetos.

Figura 2.13: Segmentación estimada a partir de bounding boxes etiquetadas.Extraído de [9]

Como todas las redes supervisadas, necesitan que los datos de *ground truth* estén etiquetados para entrenarse, en el caso de segmentación semántica es especialmente costoso ya que se necesita etiquetas por cada píxel. Entonces, hay ciertas aproximaciones para el entrenamiento de

redes semánticas que se basan en entrenamiento poco supervisado [9] que cumplen la segmentación semántica utilizando bounding boxes ya anotadas, y en algunos casos, etiquetas a nivel de imagen. Un ejemplo de ello se puede ver en la Figura 2.13.

En este trabajo, para realizar la segmentación semántica se usa una implementación de PyTorch de Deeplab, con un código original obtenido de GitHub [48]. Se pueden utilizar distintos datasets, como VOC 12, COCO o COCO-Stuff [49]. En función, del dataset utilizado los resultados variarán. En este caso, se ha utilizado COCO-Stuff 10k, que está actualizado a 20k iteraciones. Los índices de las etiquetas van de 0 a 181 y el modelo produce una distribución categórica de 182 dimensiones, pero sólo 171 clases son supervisadas con COCO-Stuff. Las imágenes de salida tienen el mismo tamaño que las de entrada, pero se puede cambiar por si se quiere reescalar en algún momento. El modelo que se utiliza es DeepLab v2 with ResNet-101 backbone, aunque se podría utilizar DeepLab v3/v3+.

2.5. Herramienta PanoContext

Con el objetivo de hacer la reproyección de imágenes, donde se harán transformaciones de coordenadas y superposición de imágenes, se utiliza la herramienta PanoContext [10] en MATLAB, que se trata de la creación de un modelo de contexto 3D para el reconocimiento de la escena panorámica. Se usan imágenes panorámicas de 360º para la reconocimiento de las escenas, generando hipótesis 3D basadas en las restricciones contextuales y clasificando las hipótesis de forma holística, llegando a un modelo de contexto 3D. De todas las herramientas que ofrece PanoContext, se utiliza la de PanoBasic, que es una toolbox para el procesamiento de imágenes panorámicas. Esta toolbox ofrece funciones de para manipular imágenes 3D, proyecciones entre la imagen panorámica y la de perspectiva normal, reconstrucción básica en 3D o procesamiento básico de la imagen como puede ser la detección de segmentos de línea, la segmentación de la imagen en color, etc. De todas ellas, se utilizan principalmente dos:

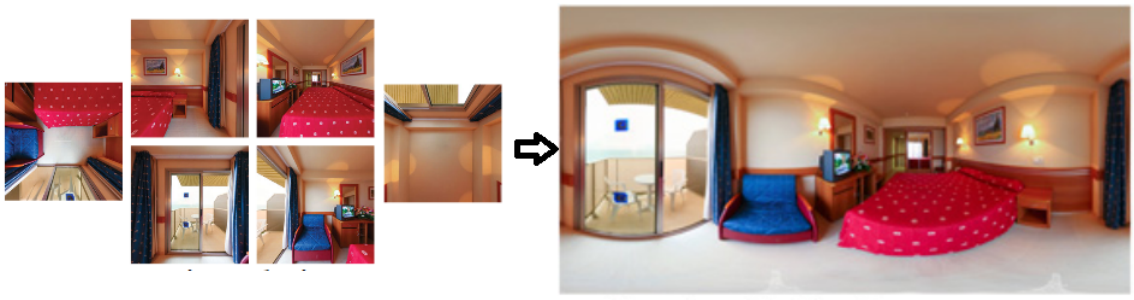


Figura 2.14: Ejemplo de Proyección entre vistas de perspectiva y panorámica. Adaptado de [10]

- **Transformación de coordenadas**: Partiendo de que una imagen panorámica es un mapa de textura en una esfera y que cada píxel de la imagen corresponde a un punto en una esfera unitaria, así como a una dirección de visión en el espacio tridimensional. Así que un píxel puede ser representado por su ubicación en la imagen(X,Y), la dirección 3D(U,V), o la posición en la esfera(x,y,z).
- **Proyección entre vistas de perspectiva y panorámica**: Una imagen panorámica puede ser proyectado a una serie de imágenes de perspectiva normal. Cada vista en perspectiva está definida por una vista en 3D en dirección d, una distancia focal R, y el tamaño de la

imagen S. El plano de la imagen correspondiente es entonces un plano en 3D que corta una esfera con radio como R en el punto R*d de la esfera. Disparando rayos desde el centro de la esfera que pasan a través de la esfera y el plano de corte, se construye la correspondencia entre la imagen panorámica y la vista en perspectiva. Un ejemplo de está proyección se puede observar en la Figura 2.14.

3

Diseño y desarrollo

3.1. Introducción

En este capítulo se verá la implementación de las herramientas necesarias para formar el marco de trabajo para segmentación semántica, mostrando primeramente una visión global del sistema mediante un esquema representativo y desglosando cada fase del esquema en las distintas secciones posteriores.

3.2. Esquema del sistema

Con el fin de desarrollar el marco de trabajo para segmentación semántica se desarrolla el sistema cuyo esquema con las distintas fases se puede observar en la Figura 3.1. Este sistema tiene como prioridad el uso de reproyecciones de imágenes en la esfera para agregar información redundante de los diferentes puntos de vista, además del uso de diferentes métodos de agregación. El sistema se puede dividir en cinco bloques. En el primer bloque, se utiliza la herramienta de la Sección 2.3 con el fin de obtener una serie de imágenes de una misma ubicación con distintos valores de heading y pitch introducidos por parámetros. El segundo bloque, la función que tiene es segmentar semánticamente las imágenes extraídas en el bloque anterior, obteniendo la máscara semántica, el labelmap ó mapa de clases y el scoremap ó mapa de confianza de cada clase. Este proceso se automatiza con la herramienta que se explica más adelante, concretamente en la Sección 3.3. El tercer bloque tiene la función de superponer las máscaras semánticas, con sus respectivos valores de heading y pitch, en un sola, con un valor de heading y pitch marcado como objetivo. Este proceso se denomina como reproyección y se desarrolla en la Sección 3.4. La reproyección implica que un píxel de la máscara semántica objetivo pueda ser apuntado por varias clases. Resolver esta disyuntiva es la función del bloque cuatro, denominado como proceso de agregación y desarrollado en la Sección 3.5. Por último, obtendremos la máscara semántica final con la que se podrá estudiar el funcionamiento de este sistema.

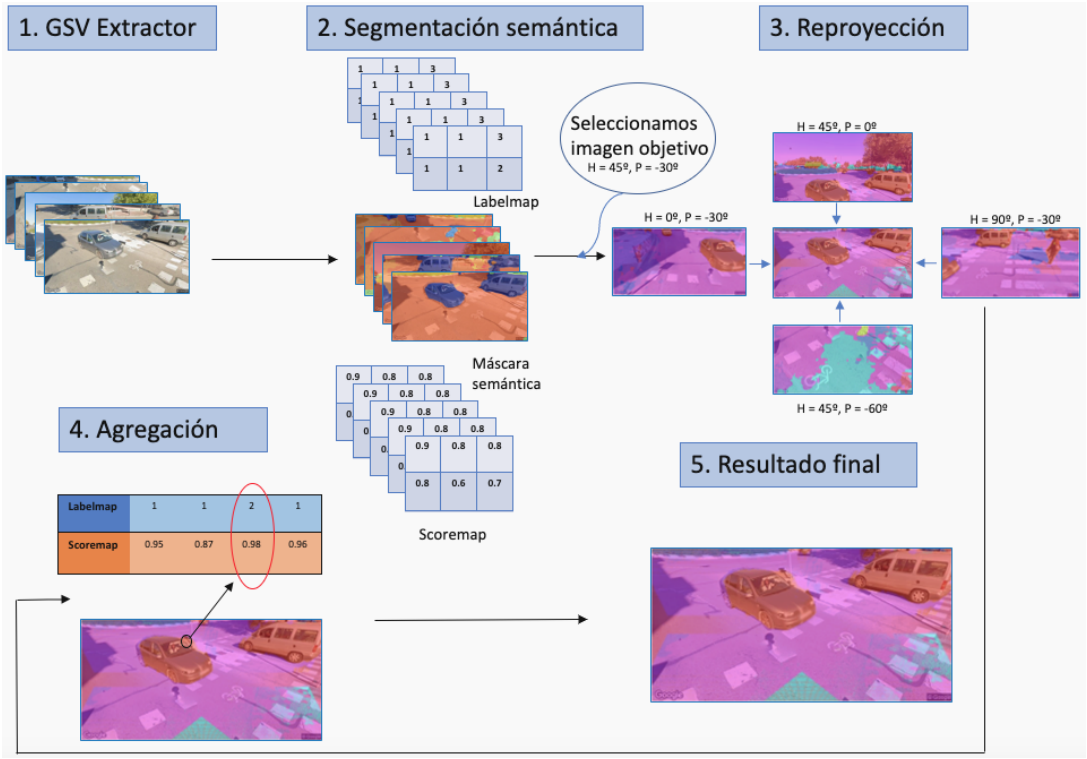


Figura 3.1: Esquema del sistema desarrollado.

3.3. Herramienta de automatización de segmentación semántica de imágenes de Street View

Figura 3.2: Ejemplo del procesado hasta la segmentación semántica de la herramienta desarrollada.

La segmentación semántica es una de las tareas principales dentro del marco del trabajo de este trabajo por lo que a partir de la implementación de PyTorch de Deeplab, con un código original obtenido de GitHub [48] se hace una serie de cambios para realizar la segmentación de semántica a las bases de datos que se obtienen de Street View Extractor. Originalmente, la herramienta sólo devolvía la máscara semántica, a lo que se añade para que también devuelva el labelmap ó mapa de clases y el scoremap ó mapa de confianza de cada clase que serán de gran utilidad e indispensables para las tareas siguientes. Un ejemplo gráfico del resultado de esta herramienta se puede observar en la figura 3.2, en el que partiendo de unas coordenadas y características, se obtiene la segmentación semántica.

3.4. Reproyección

Con el fin de poder agregar la información redundante de los diferentes puntos de vista de una ubicación se hace uso de la reproyección. Una imagen, que corresponde a una imagen tangente a la esfera con un determinado valor de heading y pitch, puede transformar sus coordenadas si el heading y pitch del centro de la imagen varía. Por tanto, este marco de trabajo tiene la capacidad de redefinir coordenadas de un heading y pitch a otro. Las imágenes pueden superponerse para formar una sola imagen con la información gráfica del resto de imágenes, esta superposición viene marcada por los valores de heading y pitch, un ejemplo gráfico se puede ver en el Figura 3.1, en el apartado de reproyección se puede observar la superposición de las máscaras semánticas, con sus respectivos valores de heading y pitch en una imagen con un valor de heading 45º y un pitch de -30º, la imagen que está en el centro. Para extraer las imágenes se utiliza un muestreo de heading y pitch habilitando la superposición de las imágenes extraídas. Este muestreo no puede ser cualquiera, debe tener un rango de valores en función del heading y pitch definido como objetivo. Por tanto, el rango de valores de heading debe determinarse como: heading máximo es igual al heading objetivo más 90º, mientras que el heading mínimo es igual al heading objetivo menos 90º ,y el rango de valores de pitch debe precisarse como: pitch máximo es igual al pitch objetivo más 90º, entretanto el pitch mínimo es igual al pitch objetivo menos 90º. La superposición de imágenes fuera de este rango traería problemas de compatibilidad y se obtendrían máscaras semánticas sin sentido. El proceso de conversión imagen-esfera-imagen es el siguiente: El primer

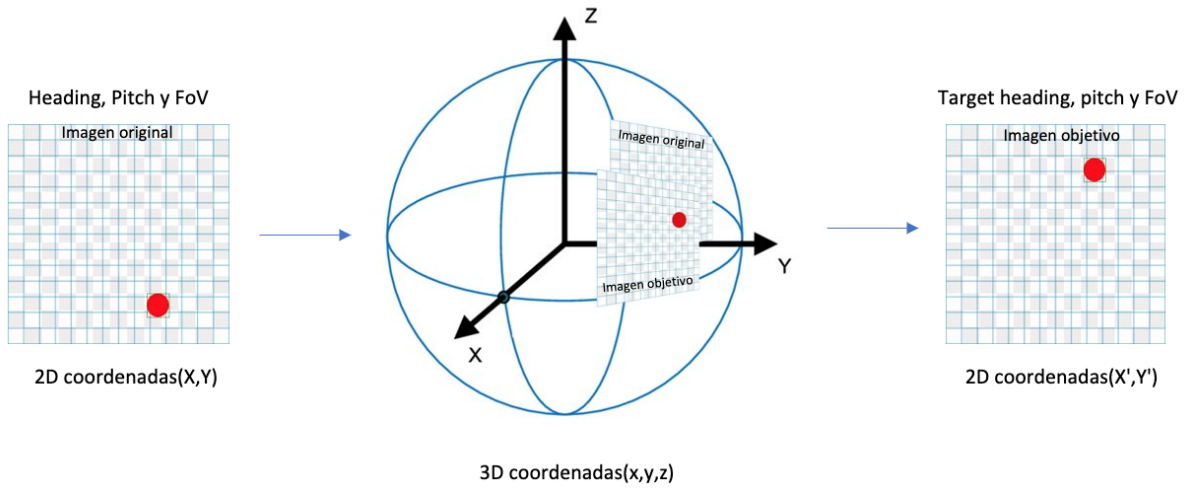


Figura 3.3: Esquema del proceso de reproyección sobre un píxel de ejemplo, marcado en rojo

paso, es hacer una proyección inversa de los píxeles de las imágenes, obtenidas del segmentador semántico, en 2D a la posición en la esfera, pasando de coordenadas (X,Y) a (x,y,z). Todo esto en función del heading, pitch, FoV, y el tamaño de la imagen. A continuación se hace la superposición de cada máscara semántica en una en función del heading y pitch marcado como objetivo. En último lugar, tendríamos una proyección de los puntos 3D(x,y,z) a coordenadas 2D(X',Y') de la imagen objetivo. Este proceso se puede observar fácilmente en la Figura 3.3, donde el punto rojo equivale a un píxel durante los 3 estados que pasa.

3.5. Agregación

Clase	1	1	43	1	43	43	27	43	1
Score	0.93	0.95	0.96	0.94	0.92	0.97	0.98	0.96	0.89

Figura 3.4: Ejemplo de vector con el labelmap y scoremap de un píxel reproyectado.

En el proceso anterior, muchas de las reproyecciones de los píxeles caen en un mismo píxel de la imagen objetivo generando superposición de clases en los píxeles. Para evitar esto se forma un vector con las clases que caen en cada píxel. El vector está formado por la clase y la confianza de dicha clase como se puede observar en la Figura 3.4. A cada uno de los elementos del vector se les denomina votantes.

La correcta o errónea decisión en la elección de la clase del vector hará que se obtengan mejores o peores resultados. Este proceso de decisión se denomina de agregación, las funciones de la misma toman varios valores de entrada y devuelve un solo resultado. Hay muchas funciones de agregación, con las que se pueden obtener múltiple valores como el valor medio, el valor máximo o mínimo, la mediana, el coeficiente de correlación, la desviación estándar, etc. Pero para este trabajo se han desarrollado los siguientes:

- **Máximo puntual**: Se escoge la clase con un valor máximo, independientemente del número de clases que haya del mismo tipo o si hay más de otro tipo como se observa en la Figura 3.5 en la que la clase resultante es la 27 con un 0.98 de confianza. Operación realizada sobre el vector de la Figura 3.4. Se denomina puntual porque sólo se tiene en cuenta la información del píxel de interés.

Clase	1	1	43	1	43	43	27	43	1
Score	0.93	0.95	0.96	0.94	0.92	0.97	0.98	0.96	0.89

Figura 3.5: Ejemplo del vector tras el método de máximo puntual.

- **Máximo ponderado puntual**: Se obtiene la suma de la confianza de todos los valores de la misma clase y se escoge el máximo de ese subconjunto como se observa en la Figura 3.6 en la que la clase resultante es la 43 con un 3.81 de confianza general del subconjunto. Operación realizada sobre el vector de la Figura 3.4.

Clase	1	43	27
Score	3.71	3.81	0.98

Figura 3.6: Ejemplo del vector tras el método de máximo ponderado puntual.

- **Máximo ponderado local**: En este método no se tiene en cuenta solo las votaciones que hay en el propio píxel bajo estudio, sino que también se tienen en cuenta las votaciones de su entorno, es decir, de una máscara alrededor de ese píxel de MxM píxeles. La máscara sigue una ponderación gaussiana, por lo que el peso del píxel central, bajo estudio, es mayor que los de alrededor. Tras estos pasos, se juntan todos los votantes en el mismo vector y se aplica el máximo ponderado puntual para obtener la clase definitiva como se observa en la Figura 3.7.

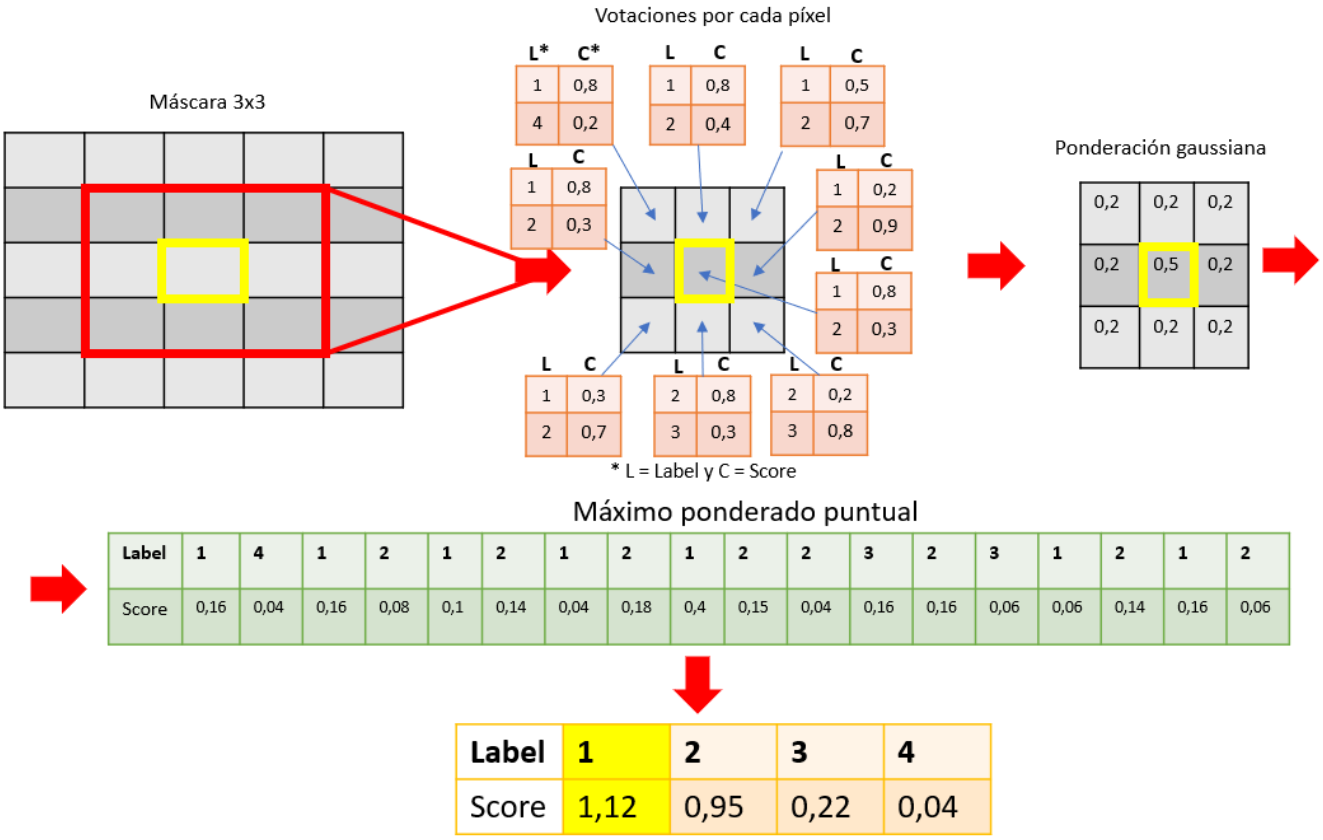


Figura 3.7: Ejemplo del proceso de agregación por máximo ponderado local con una máscara de 3x3.

4

Experimentos y resultados

4.1. Introducción

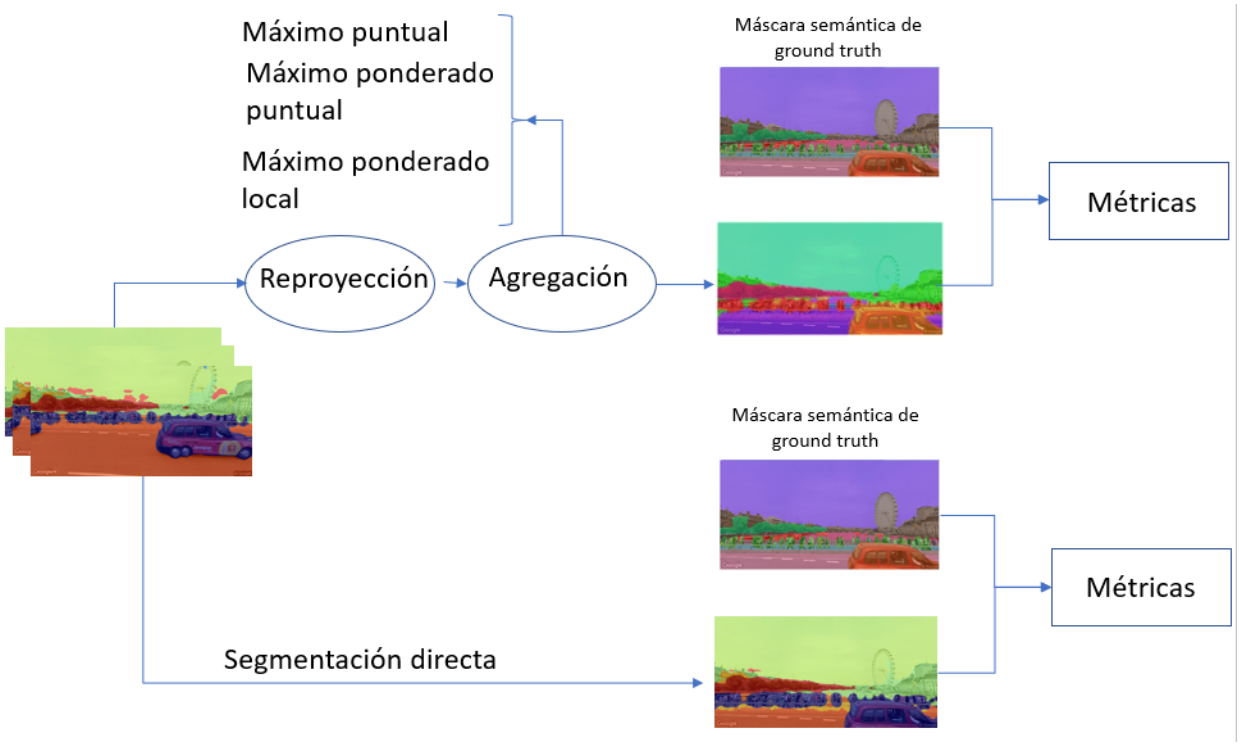


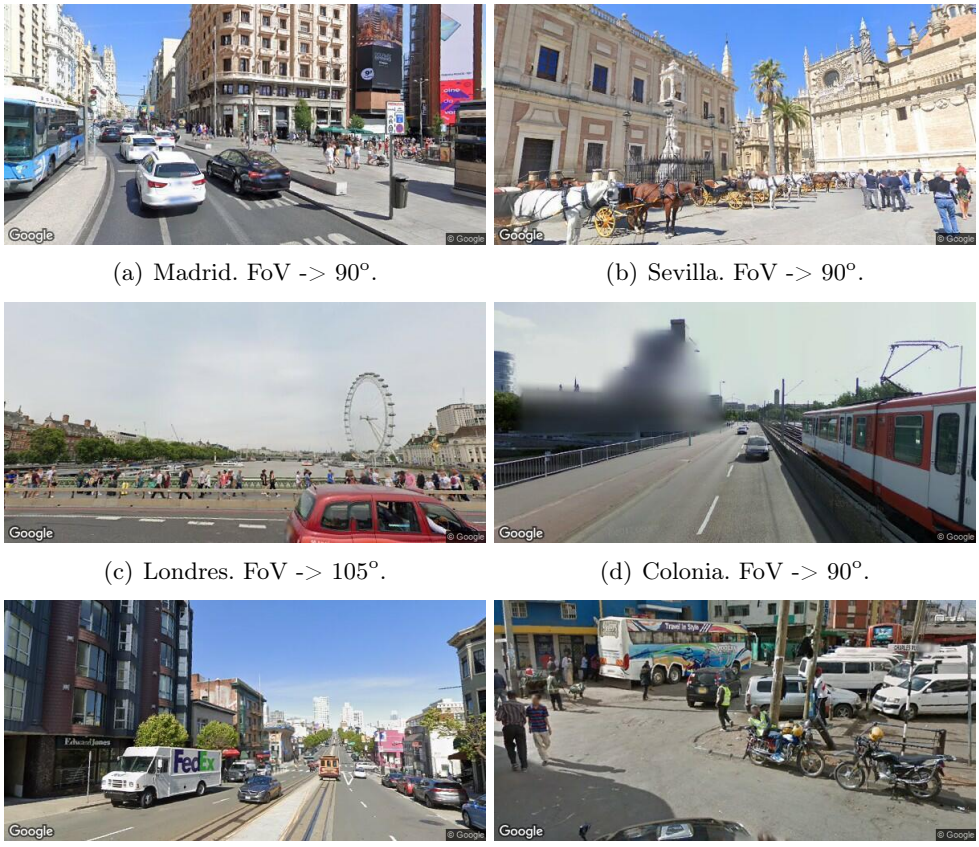
Figura 4.1: Proceso comparativo para obtener resultados entre la segmentación directa vs segmentación con información redundante de los diferentes puntos de vista. La relación entre los colores y las clases en cada imagen es distinta. Por ejemplo, la clase cielo en la imagen de *ground truth* aparece como morada, en segmentación directa verde y en reproyección y agregación como azul cían.

En este capítulo se va establecer un marco de evaluación para la segmentación semántica obtenida con diferentes métodos como se puede observar en la Figura 4.1. También se evalua-

rán las diferencias principales entre segmentación directa y segmentación por reproyección más agregación, para todos las clases de objetos y para diferentes tipos de objetos, como la división entre objetos fijos y móviles, ya que como se comentó en la introducción es muy útil para navegación automática, particularizando los resultados en ambos subconjuntos. Por último, se hará una evaluación preliminar de los diferentes métodos de agregación propuestos y la influencia del muestreo de heading y pitch en los mismos, es decir, cuánto influye la cantidad de imágenes redundante en el resultado.

4.2. Dataset

Para desarrollar los experimentos se han elegido 6 ubicaciones, en distintas partes del mundo con diferentes propiedades, sobre el que aplicar el algoritmo. Las imágenes se han seleccionado por la variabilidad de tipos de objetos, la variabilidad del número de objetos según la secuencia y la variabilidad del tamaño de los objetos. Los conjuntos de imágenes se obtienen de la herramienta de Street View Extractor.



(a) Madrid. FoV -> 90º.

(b) Sevilla. FoV -> 90º.

(c) Londres. FoV -> 105º.

(d) Colonia. FoV -> 90º.

(e) San Francisco. FoV -> 90º.

(f) Nairobi. FoV -> 70º.

Figura 4.2: Imágenes elegidas como objetivo.

Como uno de los objetivos es la influencia de la cantidad de los datos, de cada localización se toman tres muestras de distintas densidades en función de la variación del heading y pitch. Todas las localizaciones tienen una misma variación de heading y pitch, (40º,20º), (20º,10º) y (10º,5º), formando una base de datos de 90, 361 y 1369 imágenes respectivamente. Se mantiene la relación 2 a 1 entre ambos ángulos descrita anteriormente, esta misma variabilidad permitirá poder comparar las distintas localizaciones. El tamaño de las imágenes es el mismo para todos

los casos, 300x600, que es el valor por defecto de la herramienta de extracción de imágenes. De todas las imágenes de las bases de datos se define una como objetivo por localización, éstas se pueden observar en la Figura 4.2. En cada título de la imagen está anotado el FoV de la imagen, que es el mismo para cada imagen de las bases de datos de la misma localización.

(a) *Ground truth* Madrid. (b) *Ground truth* Sevilla.

(c) *Ground truth* Londres. (d) *Ground truth* Colonia.

(e) *Ground truth* San Francisco. (f) *Ground truth* Nairobi.

Figura 4.3: Imágenes *ground truth* elegidas como objetivo.

De todas estas imágenes objetivo se realiza su *ground truth* como se puede observar en la Figura 4.3, que se trata del etiquetado de las imágenes por parte de un humano, con lo que podremos comparar los resultados del segmentador y del algoritmo implementado.

4.3. Métricas de evaluación

Para comparar los distintos escenarios se utilizan diferentes medidas que se definen a continuación. Todos los escenarios se comparan con máscaras semánticas de *ground truth*. Una tabla de clasificación binaria define cuatro medidas:

- **Verdadero positivo(TP)**: La clase real y la detectada son ambas positivas.
- **Falso positivo(FP)**: La clase real es negativa pero la clase detectada es positiva.
- **Verdadero negativo(TN)**: La clase real y la detectada son ambas negativas.
- **Falso negativo(FN)**: La clase real es positiva pero la clase detectada es negativa.

Este trabajo no es binario, es un problema multiclase, por lo que el significado de los conceptos desarrollados anteriormente se extiende a multiclase. Con estas medidas se definen los siguientes

cálculos utilizados para evaluar la segmentación semántica. La medida de ***Accuracy*** (4.1) es quizás la más fácil de entender conceptualmente ya que no es más que el porcentaje de píxeles de su imagen que se clasifican correctamente. El problema de esta medida es cuando se encuentra ante imágenes con desequilibrio de clases, que resulta poco útil.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{4.1}$$

La medida de ***Intersection over union (IoU)***, también conocida como índice de Jaccard, es la métrica más utilizada para segmentación semántica [7]. La IoU es el área de superposición entre la segmentación predicha y el *ground truth* dividida por el área de unión entre la segmentación predicha y el *ground truth* (4.2).

$$IoU = \frac{TP}{TP + FP + FN} \tag{4.2}$$

La medida de ***Weighted IoU*** se trata de la media ponderada entre la medida de IoU por clase y el número de píxeles de dicha clase. La medida de ***Precision*** se define como la capacidad de etiquetar negativamente una muestra que no es positiva (4.3).

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

El ***Recall*** (4.4) es la capacidad para detectar todas las muestras positivas.

$$Recall = \frac{TP}{TP + FN} \tag{4.4}$$

Estas dos últimas medidas se utilizan para calcular el ***F1-Score*** (4.5), también conocido como coeficiente de Dice, que se podría interpretar como una medida de rendimiento global del sistema.

$$F1 - Score = \frac{2 * Recall * Precision}{Recall + Precision} \tag{4.5}$$

A partir de la anterior, se obtiene la métrica de ***BFScore***, que es una medida de ***F1-Score*** calculada en los contornos, por tanto indica lo bien que se alinea el límite previsto de cada clase con el límite real. Para facilitar las comparaciones, se utilizan el promedio de las distintas para los valores anteriores, estos se denominan ***mean Accuracy***, ***mean IoU*** y ***mean BFScore***. También se utiliza la medida de ***global Accuracy***, que es la relación entre los píxeles correctamente clasificados, sin importar la clase, y el número total de píxeles.

4.4. Experimentos

En esta sección se van a exponer los experimentos que se realizan y sus resultados, teniendo en cuenta que los objetivos de evaluación son los siguientes:

- Agregación frente a la segmentación directa.
- La comparación entre agregaciones.
- Los resultados en las diferentes clases.
- Los resultados entre las clases de objetos móviles y fijos.
- La influencia del muestro.

Los experimentos se realizan en MATLAB, utilizando la toolbox de computer vision.
Al contar con un dataset en el que muchas de las clases representan conceptos muy detallados que para el propósito de este trabajo, se pueden generalizar, por lo que se decide unificar las clases de stuff a un conjunto de clases menor que tengan relación o parecido, en este caso, se unifican al conjunto con nombre en la posición anterior siguiendo la estructura de la Figura 2.11, con lo que se consigue una reducción del número de clases y, por tanto, facilitar el trabajo, puesto que una diferenciación tan fina no es necesaria, y puede distorsionar los resultados obtenidos . Un par de ejemplos para dar claridad sería que las clases *water-other, waterdrops, sea, river y fog* se unifican en un conjunto denominado *water* o que las clases *building-other, roof, tent, bridge, skyscraper y house* en *building*. También se hace una división entre objetos móviles y fijos para comparar resultados.

4.4.1. Resultados experimentales

En esta sección se mostrarán los distintos experimentos y sus objetivos, aquello que se quiere evaluar, los resultados que se esperan y por último, los resultados y un análisis de los mismos. Los resultados se muestran en porcentaje decimal y para el método de máximo ponderado local se utiliza un filtro predefinido con distribución gaussiana de 3x3.

Experimento rendimiento global

Este experimento pretende dar una visión global de los resultados del marco de trabajo desarrollado. Tiene varios objetivos de evaluación, el primero es comparar la agregación frente a la segmentación directa, el segundo es la comparación entre agregaciones y por último, la influencia del muestreo. En la Tabla 4.1 se pueden ver los resultados de los distintos tipos de agregación y segmentación semántica directa en función de la variación de heading y pitch, es decir, el muestreo. Estos resultados son obtenidos como promedio de las distintas localizaciones.

Variación H/P	Método	Global Accuracy	Mean Accuracy	Mean IoU	Weighted IoU	Mean BFScore
Variación 40/20	Segmentador semántico	**0,661**	**0,483**	0,254	**0,583**	**0,526**
	Max puntual	0,647	0,470	0,202	0,554	0,459
	Max ponderado puntual	0,642	0,463	0,206	0,554	0,446
	Max ponderado local	0,642	0,464	0,230	0,555	0,464
Variación 20/10	Segmentador semántico	**0,661**	**0,483**	0,254	**0,583**	**0,526**
	Max puntual	0,634	0,455	0,232	0,539	0,432
	Max ponderado puntual	0,643	0,466	0,259	0,557	0,473
	Max ponderado local	0,644	0,466	0,268	0,557	0,469
Variación 10/5	Segmentador semántico	**0,661**	**0,483**	0,254	**0,583**	**0,526**
	Max puntual	0,628	0,451	0,204	0,531	0,423
	Max ponderado puntual	0,644	0,468	0,267	0,558	0,470
	Max ponderado local	0,645	0,468	**0,270**	0,558	0,485

Cuadro 4.1: Resumen de resultados de los promedios de las métricas de todas la localizaciones.

A priori, se podría pensar que la densidad de imágenes de la base de datos sería bastante determinante en los resultados, ya que a mayor número de imágenes mayor número de votantes y por tanto, mejores resultados pero como se puede observar, las tres variaciones de heading y pitch muestran resultados muy similares. Por ejemplo, la media de *Weighted IoU* para el método de máximo ponderado local es 55.5 % para la variación 40/20, 55.7 % para la variación 20/10 y 55.8 % para la variación 10/5. En el resto de métricas y métodos, muestran un línea continuista

en la que la variación con mayor densidad se obtienen mejores resultados, pero unas diferencias muy pequeñas y poco determinantes como refleja la Tabla 4.1.

Localización	Variación Heading 40º Varación Pitch 20º	Variación Heading 20º Varación Pitch 10º	Variación Heading 10º Varación Pitch 5º
Madrid	6,65	25,36	99,95
Sevilla	6,81	25,89	102,99
Londres	11,15	40,15	167,89
Colonia	6,66	25,37	100,16
San Francisco	6,81	25,89	102,99
Nairobi	3,71	14,24	55,75
Promedio	6,96	26,15	104,95

Cuadro 4.2: Resultados del número medio de votantes por píxel por cada variación en las distintas localizaciones.

Localización	Variación Heading 40º Varación Pitch 20º	Variación Heading 20º Varación Pitch 10º	Variación Heading 10º Varación Pitch 5º
Madrid	1,52	1,83	2,33
Sevilla	1,29	1,53	2,13
Londres	1,9	2,41	3,57
Colonia	1,67	2,39	3,37
San Francisco	1,32	1,69	2,37
Nairobi	1,57	2,33	3,09
Promedio	1,54	2,03	2,81

Cuadro 4.3: Resultados del número medio de clases votantes por píxel por cada variación en las distintas localizaciones.

Con el fin de intentar entender la invariabilidad de los resultados en función del muestro se obtienen dos tablas de resultados, la primera, la Tabla 4.2, que muestra el número medio de votantes por píxel por cada variación en las distintas localizaciones, y un promedio de todas ellas. Se puede observar una gran diferencia entre los muestreos, la variación menos densa tiene un ratio 1:4 frente a la de densidad media y un ratio 1:15 frente a la variación más densa, por lo que por cada votante de la variación menos densa hay 15 de la más densa, lo que es una diferencia muy reseñable. Y la segunda, la Tabla 4.3, que muestra el número medio de clases votantes por píxel por cada variación en las distintas localizaciones, y un promedio de todas ellas. En este caso la variación es bastante pequeña, con un ratio 1:1.33 entre la variación menos densa y la de densidad media, mientras que el ratio entre variación menos densa y la más densa es de 1:1.82. Esto último refleja que utilizar la agregación es útil al contar con un número muy corto de clases, entre 1 y 3, si este número fuese muy superior utilizar la agregación no valdría para nada. Por tanto, que no haya una mejoría en función del muestreo, visto que hay suficientes votantes por píxeles, puede ser consecuencia de la distribución de scores en los votantes. Esto sería interesante seguir investigarlo y por eso, se propondrá como posible trabajo futuro.

(a) Max Puntual. (b) Max Ponderado Puntual . (c) Max Ponderado Local.

Figura 4.4: Máscaras semánticas de Londres mostrando las diferencias entre los métodos de agregación.

Una vez tratado el objetivo de la influencia de muestreo, se procede a evaluar la comparación entre los distintos métodos de agregación. En un principio, se prevé que usando métodos de agregación más complejos se obtendrán mejores resultados. En la Tabla 4.1 se puede observar los resultados de las métricas de los tres métodos de agregación utilizados, el máximo puntual, el máximo ponderado puntual y el máximo ponderado local, refutando lo que se intuía pero con unas diferencias entre los métodos bastante corta, especialmente entre el máximo ponderado puntual y el máximo ponderado local, que como por ejemplo, en *Mean BFScore* donde se encuentra la máxima diferencia entre ambos métodos es de 47 % contra 48.5 % en la variación más densa, mientras que en otras métricas como *Mean Accuracy* o *Weighted IoU* se obtiene el mismo resultado, 46.8 % y 55.8 % respectivamente, también en la variación más densa. Un ejemplo gráfico de las diferencias entre los distintos métodos de agregación se puede observar en la Figura 4.4, en el que se muestra la localización de Londres con un muestreo de densidad media. Se ha escogido este ejemplo porque refleja lo descrito anteriormente, el método de máximo puntual es peor que el máximo ponderado puntual y el máximo ponderado local, que son muy parejos. En la subimagen a, se observa como en la categoría cielo, pintada en color azul, hay también color rosa, el cual está asociado a la clase carretera. En la subimagen b y la c, al ser métodos algo más complejos se observa que este error desaparece etiquetando correctamente la clase cielo en ambas máscaras semánticas, en la subimagen b pintada en verde ,y en la subimagen c, en azul.

Por último, se procede a evaluar la comparación entre la segmentación directa, y los distintos métodos de agregación. Los resultados en negrita de la Tabla 4.1 muestran los valores máximos por cada métrica. Estos reflejan que con la segmentación directa se obtienen mejores resultados en las métricas bajo estudio salvo en la de *Mean IoU* donde la diferencia entre la segmentación directa y el método de agregación de máximo ponderado puntual, el que mejores resultados ofrece de los métodos de agregación, es de 25.4 % frente a 27 % en la variación más densa.

En definitiva, al tener un conjunto de datos muy reducido las conclusiones que se pueden obtener tienen limitaciones, pero se podría concluir que la influencia del muestreo es poco determinante en los resultados, aunque es necesario seguir profundizando en esta línea. También se podría concluir que con los métodos de agregación implementados la segmentación directa es mejor en todas las medidas menos en la *Mean IoU*, por lo que sería interesante aumentar la sofisticación de los métodos de agregación para comprobar si el resto de métricas pueden llegar a superar a la segmentación directa.

Experimento de objetos móviles y fijos

El contexto de este trabajo es la navegación automática, donde la división entre objetos móviles y fijos es de gran utilidad para la localización y guiado como se comentaba en la introducción de este trabajo. En especial, los objetos fijos son más determinantes para la localización porque suelen ser atemporales, por esta razón se decidió dividir las clases en objetos móviles y fijos de cada localización. Por tanto, se presentan los resultados de esta división cumpliendo uno de los objetivos presentados al inicio de esta sección. A partir de este momento, como la densidad de las bases de datos no es determinante y con el fin de mostrar datos más concretos, se decide utilizar una densidad de datos media, es decir, con una variación de heading de 20º y de pitch de 10º.

Métrica	Experimento	Objetos móviles	Objetos fijos
Accuracy	Segmentador semántico	**0,551**	**0,477**
	Max puntual	0,529	0,445
	Max ponderado puntual	0,528	0,463
	Max ponderado local	0,529	0,464
IoU	Segmentador semántico	**0,374**	0,250
	Max puntual	0,290	0,234
	Max ponderado puntual	0,312	0,278
	Max ponderado local	0,314	**0,283**
BFScore	Segmentador semántico	**0,551**	**0,527**
	Max puntual	0,447	0,445
	Max ponderado puntual	0,467	0,487
	Max ponderado local	0,464	0,483

Cuadro 4.4: Resumen de resultados de los promedios de las métricas de todas la localizaciones divididas las clases en objetos fijos y móviles.

En la Tabla 4.4 se pueden observar dichos resultados, en negrita aparecen los valores máximos por cada métrica. Para analizar los resultados, se divide el estudio en los objetos móviles por un lado y los fijos por otro, para comparar ambos al final. Se procederá a hacer una comparación entre los métodos de agregación y estos con la segmentación directa.

En el caso de los objetos móviles, y en referencia a la comparación entre los distintos métodos de agregación se puede observar que para la métrica de *Accuracy* apenas hay diferencia entre los tres métodos, un 0.1 %. En el caso de *IoU*, hay más diferencia, el máximo ponderado local es 2.9 % mejor que el máximo puntual y sólo un 0.2 % mejor que el máximo ponderado puntual. En la métrica de *BFScore*, ocurre algo similar que en la anterior, pero en este caso el método de máximo ponderado puntual es con el que se obtienen mejores resultados, un 2 % mejor que el máximo puntual y un 0.3 % que el máximo ponderado local. Analizando los métodos de agregación que mejores resultados ofrecen frente a la segmentación directa se puede ver un claro ganador, la segmentación directa, que es un 2.2 % mejor en *Accuracy*, un 6 % mejor en *IoU* y un 8.4 % mejor en *BFScore*.

Ahora, es el turno de los objetos fijos, donde se hará un análisis similar al anterior. Para la métrica de *Accuracy* el máximo ponderado local es 1.9 % mejor que el máximo puntual y sólo un 0.1 % mejor que el máximo ponderado puntual. Para la métrica de *IoU* ocurre algo similar, el máximo ponderado local es 4.9 % mejor que el máximo puntual y un 0.5 % mejor que el máximo ponderado puntual. En la última métrica, el escenario es bastante parecido al de los objetos móviles en el cual el método de máximo ponderado puntual es con el que se obtienen mejores

resultados, un 4.2 % mejor que el máximo puntual y un 0.4 % que el máximo ponderado local. En este caso, al compararlos frente a segmentación directa se observa que en *IoU* la agregación ofrece mejores resultados, 3.3 % mejor. En el resto de métricas la segmentación directa gana, 1,3 % en *Accuracy* y un 4 % en *BFScore*.

Comparando objetos móviles y fijos, se obtienen mejores resultados en los móviles en las tres métricas, un 7.4 % mejor en *Accuracy*, un 9.1 % mejor en *IoU* y un 2.4 % mejor en *BFScore*, pero lo que cabe destacar es que mientras en los objetos fijos las diferencias entre los métodos de agregación y la segmentación directa es pequeña, incluso, en *IoU* es superior la agregación, en los objetos móviles hay mucha diferencia entre la segmentación directa y la agregación. Esto explicaría que en las métricas de la Tabla 4.1, la única métrica en la que era superior la agregación era de *Mean IoU*. Por tanto, en localizaciones donde haya mayor número de objetos fijos la agregación será mejor método para obtener mejores resultados de *IoU*.

Pese a la limitación de las conclusiones por la pequeña cantidad de datos se podría definir que los objetos móviles son segmentados con mayor precisión mientras que los objetos fijos muestran peores resultados pero reflejan que la agregación es útil para los mismos. Por lo que sería interesante analizar con más datos y métodos de agregación más sofisticados con el fin de observar cómo afecta tanto a objetos fijos como móviles. Si en el caso de los fijos, esa sofisticación podría permitir tener unos resultados que se acerquen a los móviles, y en el caso de los móviles, si la sofisticación pudiera mejorar a la segmentación directa.

Experimento de clases

Ante la vista de la diferencia entre los objetos fijos y móviles y con el fin de profundizar en observar que clases se obtienen mejores resultados, se ha decidido analizar todas la clases existentes en los 6 escenarios y sacar las métricas de cada uno con el fin de cumplir el objetivo evaluar los resultados en las diferentes clases descrito al principio de la sección. Se obtienen tres tablas , una por cada métrica, *Accuracy*, *IoU* y *BFScore*, mostrando los resultados de cada clase en función del método de agregación o segmentación directa. Los valores que son 0 corresponden al caso en el que la etiqueta está en el *ground truth* pero no ha aparecido en la segmentación semántica.
La primera que se va a estudiar es la métrica *Accuracy*, a priori, analizando los resultados del experimento anterior se prevé que en la mayoría de clases se obtengan mejores resultados por segmentación directa tanto en las clases que pertenecen a objetos móviles como fijos.

Accuracy				
Clase	Segmentador semántico	Max Puntual	Max Ponderado Puntual	Max Ponderado Local
person	**0,587**	0,540	0,582	0,584
bicycle	**0,049**	0,000	0,000	0,000
car	**0,715**	0,711	0,696	0,696
motorcycle	**0,450**	0,419	0,442	0,442
bus	**0,912**	0,761	0,805	0,806
train	**0,500**	0,000	0,000	0,000
truck	**0,413**	0,259	0,322	0,320
boat	0,000	**0,189**	0,048	0,048
traffic_light	0,000	0,000	0,000	0,000
street_sign	0,000	0,000	0,000	0,000
bench	0,000	0,000	0,000	0,000
dog	0,000	0,000	0,000	0,000
horse	0,742	**0,777**	0,753	0,753
suitcase	0,000	0,000	0,000	0,000
building	0,683	0,796	**0,811**	**0,811**
vegetation	0,700	0,676	0,720	**0,721**
sky	**0,603**	0,585	0,588	0,588
water	0,000	0,024	**0,026**	0,024
solid	0,000	0,000	0,000	0,000
structural	**0,094**	0,000	0,000	0,000
ground	**0,475**	0,371	0,398	0,399

Cuadro 4.5: Resultados de accuracy por clase de todas las localizaciones.

En la Tabla 4.5 se obtienen los resultados de la métrica de *Accuracy*, mostrando en negrita los valores máximos por clase en función del método utilizado. Como se observa, en la mayoría de clases se obtienen mejores resultados por segmentación directa, excepto en las clases *boat y horse*, que se obtiene con el método de máximo puntual, en *water* con el máximo ponderado puntual y en *building, vegetation* con el máximo ponderado local, lo que corrobora la suposición hecha con anterioridad. Las clases que se clasifican dentro de objetos móviles tienden a obtener resultados más precisos con segmentación directa, sólo hay dos excepciones, las clases *boat y horse*, mientras aquellas clases que se clasifican dentro de objetos fijos no tienen una distinción tan clara puesto que las clases *sky, ground y structural* son más precisas con segmentación directa entretanto las clases *building, vegetation y water* son más precisas con agregación.

Las diferencias entre agregación y segmentación directa no es muy significativa, por ejemplo las máximas diferencias, quitando aquellas donde en algún método la clase aparece en el *ground truth* pero no en la máscara semántica y aparece un 0 en el resultado de la tabla como es el caso de*structural o bicycle* entre otras , son de 12.7 % en la clase *building* y 10.54 % en la clase *bus*. Las mínimas diferencias son de 0.3 % en la clase *person* y 0.76 % en la clase *motorcycle*. Lo que indica que las diferencias entre la segmentación directa y la agregación en términos de *Accuracy* no son muy amplias, para terminar de ratificarlo se realiza el porcentaje medio de la diferencia entre segmentación directa y la agregación obteniendo un valor de 2.54 %.

La siguiente métrica a analizar es la de *IoU*, en principio, los resultados del experimento anterior arrojan que para los objetos fijos, la agregación será el mejor método, en particular el máximo ponderado local, mientras que para los móviles, la segmentación directa seguirá siendo la mejor. Por tanto, se prevé que las clases pertenecientes a objetos fijos tengan mejores resultados para las medidas de *IoU* por agregación y aquellas pertenecientes a móviles, mejores resultados por segmentación directa.

IoU				
Clase	Segmentador semántico	Max Puntual	Max Ponderado Puntual	Max Ponderado Local
person	0,379	0,366	**0,385**	**0,385**
bicycle	**0,043**	0,000	0,000	0,000
car	**0,517**	0,482	0,487	0,488
motorcycle	**0,346**	0,288	0,286	0,285
bus	0,379	0,659	0,475	**0,713**
train	**0,443**	0,000	0,000	0,000
truck	**0,206**	0,129	0,160	0,160
boat	0,000	**0,153**	0,046	0,046
traffic_ight	0,000	0,000	0,000	0,000
street_ign	0,000	0,000	0,000	0,000
bench	0,000	0,000	0,000	0,000
dog	0,000	0,000	0,000	0,000
horse	0,510	**0,541**	0,535	0,535
suitcase	0,000	0,000	0,000	0,000
building	0,156	0,176	**0,182**	**0,182**
vegetation	0,246	0,212	**0,459**	**0,459**
structural	**0,006**	0,000	0,000	0,000
sky	**0,220**	0,198	0,202	0,202
water	0,000	**0,024**	0,013	**0,024**
solid	0,000	0,000	0,000	0,000
ground	0,118	0,120	**0,154**	**0,154**

Cuadro 4.6: Resultados de *IoU* por clase de todas las localizaciones.

Los resultados se muestran en la Tabla 4.6, aquellos representados en negrita son los valores máximos por clase en función del método utilizado. En este caso, la tendencia no está tan definida como en el caso de *Accuracy* y está más dividida. Las clases donde se obtienen mejores resultados de *Iou* por clase en segmentación directa son *bicycle, car, motorcycle, train, truck, structural y sky*, en el resto de clases gana la agregación, concretamente las clases *boat, horse, water* con el método de máximo puntual, la clase *person, building, vegetation, ground* con el método de máximo ponderado puntual y las clases *person, bus, building, vegetation, water, ground*. Las clases que se clasifican dentro de objetos móviles tienden a obtener mejores resultados con segmentación directa, salvo en los casos de *boat, horse, bus y person*, mientras aquellas que se clasifican dentro de objetos fijos tienden a obtener mejores resultados con la agregación, a excepción de *structural y sky*. El porcentaje medio de la diferencia entre segmentación directa y la agregación tiene un valor de 0.79 %. Los casos en los cuales hay una mayor mejoría en los resultados en la agregación es en *bus*, donde mejora un 33.45 % y *vegetation* un 21.38 %. Mientras que en el caso contrario, las clases que más empeoran en la agregación frente a la segmentación directa son *truck y motorcycle* con 4.55 % y 5.81 % respectivamente.

La siguiente métrica a analizar es la de *BFScore*, en principio, los resultados del experimento anterior arrojan que para los objetos fijos y móviles, la segmentación directa será el mejor método. Por tanto, se prevé que casi todas las clases tengan mejores resultados para las medidas de *BFScore* por segmentación directa pero aquellas clases pertenecientes a objetos fijos la diferencia entre las segmentación y la agregación sea menor que la de los móviles.

BFScore				
Clase	Segmentador semántico	Max Puntual	Max Ponderado Puntual	Max Ponderado Local
person	0,563	0,511	**0,590**	0,575
bicycle	**0,303**	0,000	0,000	0,000
car	**0,617**	0,348	0,396	0,414
motorcycle	**0,705**	0,435	0,456	0,458
bus	**0,633**	0,411	0,468	0,446
train	**0,712**	0,000	0,000	0,000
truck	**0,392**	0,240	0,228	0,207
boat	0,000	**0,594**	0,451	0,435
traffic_ight	0,000	0,000	0,000	0,000
street_ign	0,000	0,000	0,000	0,000
bench	0,000	0,000	0,000	0,000
dog	0,000	0,000	0,000	0,000
horse	0,363	0,489	**0,503**	0,480
suitcase	0,000	0,000	0,000	0,000
building	0,537	0,557	0,582	**0,583**
vegetation	0,527	0,523	**0,568**	0,559
sky	0,551	0,581	0,671	**0,687**
water	0,000	0,130	**0,171**	0,103
solid	0,000	0,000	0,000	0,000
ground	**0,499**	0,359	0,496	0,494
structural	**0,252**	0,000	0,000	0,000

Cuadro 4.7: Resultados de BFScore por clase de todas las localizaciones.

Los resultados se muestran en la Tabla 4.7, aquellos representados en negrita son los valores máximos por clase en función del método utilizado. Se preveía que la mayoría de clases fuera mejor la segmentación directa, y esto es así en casi todas las clases pertenecientes a objetos móviles, excepto las clases de *boat, horse y person*. Mientras que en las clases pertenecientes a objetos fijos se obtienen mejores resultados con segmentación directa sólo en los casos de *ground y structural*, por lo que el resto de clases, *building, vegetation, sky y water* obtiene mejores resultados con métodos de agregación. El porcentaje medio de la diferencia entre segmentación directa y la agregación tiene un valor de 5.51 %. Los casos en los cuales hay una mayor mejoría en los resultados en la agregación es en *sky*, donde mejora un 13.58 % y *horse* un 13.93 %. Mientras que en el caso contrario, las clases que más empeoran en la agregación frente a la segmentación directa son *car y motorcycle* con 20.31 % y 24.63 % respectivamente.

Analizando los resultados se encuentra que hay clases que obtienen mejores resultados en las tres métricas para segmentación directa, éstas son *bicycle, car, motorcycle, train, truck y structural*. Luego están aquellas que obtienen mejores resultados en las tres métricas para agregación que son *building, vegetation, water, boat y horse*. Por otro lado, hay clases que obtienen mejores resultados en dos de las tres métricas para segmentación directa, que son *bus, sky y ground*. Y la

clase que falta, *person*, obtiene mejores resultados en dos de las tres métricas para agregación. Por eso se estudian las tres métricas, dos de ellas, *Accuracy y BFScore*, muestran mejores resultados para segmentación directa, aunque en el caso de BFScore está bastante reñido, 8 clases muestran mejores resultados con segmentación directa contra 7 que muestran mejores resultados por agregación. La métrica restante, *IoU*, devuelve mejores resultados para la agregación, en particular, el método de máximo ponderado local. Estos datos representan que los objetos móviles suelen tener mejores resultados para segmentación directa en la mayoría de métricas, mientras los objetos fijos suelen tener mejores agregación en la mayoría de las métricas, pero todas las conclusiones están limitadas por la poca cantidad de datos con los que se ha hecho el experimento.

5

Conclusiones y trabajo futuro

5.1. Conclusiones

Se ha desarrollado el marco de trabajo para segmentación semántica en imágenes de Google Street View que aprovecha las características de las bases de datos de 360º, propuesto como uno de los objetivos principales. Este marco de trabajo es útil para la navegación automática, que ayuda a la localización y guiado, donde un proceso de segmentación semántica en el cual se analizan las imágenes a nivel del píxel, puede ayudar a que funcione mejor el método de navegación.

Se llevan a cabo una serie de experimentos sobre el marco de trabajo desarrollado, los objetivos de estos experimentos son evaluar los métodos simples de agregación propuestos frente a la segmentación directa, comparar los métodos de agregación entre sí, conocer la influencia del muestreo y analizar los resultados a nivel de clase semántica, siendo también agrupadas en objetos fijos y móviles. Las conclusiones de los experimentos están limitadas por tener un conjunto de datos muy reducido.

Los métodos de agregación implementados en este trabajo son preliminares, ya que tampoco es la finalidad del mismo obtener el método de agregación más concluyente. Se muestra que no hay unos resultados concluyentes en cuanto a que se mejore la segmentación semántica obtenida por los métodos de reproyección y agregación con respecto a la segmentación directa. Por tanto, la hipótesis realizada de si con el uso de múltiples puntos de vista de una localización mejoraría la segmentación semántica que se puede obtener de un solo punto de vista que se planteaba al principio, se concluye que con los métodos de agregación vistos no drásticamente, en función de los casos por lo que sería interesante seguir investigándolo.

La densidad de muestro de las bases de datos no es relevante a la hora de obtener mejores o peores resultados, algo que a priori, se pensaba que era un factor determinante.

Las métricas utilizadas en el trabajo muestran que en todas ellas se obtienen mejores resultados con el segmentador semántico excepto la medida de IoU, que muestra mejores resultados la agregación, en concreto, el método de máximo ponderado local.

La división entre objetos fijos y móviles puede ser de gran utilidad para la localización y guiado, sobre todo, los objetos fijos, que son atemporales, puesto que los elementos móviles pueden distorsionar el proceso de aprendizaje. Los resultados permiten concluir que para los objetos fijos se obtienen mejores rendimientos con el método de agregación de máximo ponderado local, mientras que para los objetos móviles es con el segmentador semántico. Con el contexto de la navegación automática, y el interés por la correcta segmentación de los objetos fijos estas conclusiones pueden ser interesantes, puesto que una sofisticación de los métodos de agregación permitiría aumentar los resultados de la segmentación de los objetos fijos.

5.2. Trabajo futuro

Una vez creado el marco de trabajo, la idea sería seguir una línea continuista con el fin de obtener datos más concluyentes. Para ello sería necesario una evaluación sobre más bases de datos y más grandes que las utilizadas en este trabajo, con el fin de evaluar si las primeras conclusiones obtenidas son veraces.

Con las conclusiones obtenidas del paso anterior se podrían desarrollar métodos de agregación más sofisticados para mejorar los resultados e intentar superar los de segmentación directa.

Habría que seguir investigando la influencia del muestreo, para entender su invariabilidad, estudiando pormenorizado las condiciones de voto en los diferentes métodos.

En este trabajo, el proceso de fusión de máscaras semánticas, la denominada reproyección, se hace sobre múltiple imágenes y cuya finalidad es obtener una imagen final de perspectiva. Por tanto, como se dispone de la información 360º y de las herramientas para reproyectar todas las imágenes a la imagen objetivo de 360º, sería interesante evaluar si existen beneficios al hacer la reproyección y agregación sobre el dominio de la imagen panorámica. Esto podría tener interés para otras aplicaciones más allá de la navegación automática.

Glosario de acrónimos

- **GSV**: *Google Street View*
- **GPS**: *Global Positioning System*
- **API**: *Application Programming Interface*
- **FOV**: *Field of View*
- **CNN**: *Convolutional Neural Networks*
- **COCO**: *Common Objects in COntext*
- **VOC**: *Visual Object Classes*
- **SPP**: *Spatial Pyramidal Pooling*
- **MPA**: *Multi-scale Patch Aggregation*
- **FCN**: *Fully Convolutional Network*
- **PSP**: *Pyramidal Scene Parsing*
- **FPN**: *Feature Pyramid Network*
- **ASPP**: *Atrous Spatial Pyramid Pooling*
- **CRF**: *Conditional Random Field*

Bibliografía

[1] Fuentes de las fotografías. (s.f.). Recuperado de: https://www.google.com/streetview/explore.

[2] Adrian Rosebrock. Image Stitching with OpenCV and Python. Recuperado de: https://www.pyimagesearch.com/2018/12/17/image-stitching-with-opencv-and-python/, 2017.

[3] Street View Service. (s.f.). Recuperado de: https://developers.google.com/maps/documentation/javascript/

[4] Cristóbal Silva. *Modelamiento Semántico del Entorno de un Robot utilizando información RGB-D*. PhD thesis, 2016.

[5] Priya Dwivedi. Semantic segmentation — popular architectures. Recuperado de: https://towardsdatascience.com/semantic-segmentation-popular-architectures-dff0a75f39d0, 2019.

[6] Semantic Segmentation. (s.f.). Recuperado de: https://www.mathworks.com/solutions/image-video-processing/semantic-segmentation.html.

[7] COCO - Common Objects in Context. Recuperado de: http://cocodataset.org/stuff-eval.

[8] Sik-Ho Tsang. Review: Dilatednet — dilated convolution (semantic segmentation). Recuperado de: https://towardsdatascience.com/review-dilated-convolution-semantic-segmentation-9d5a5bd768f5, 2018.

[9] George Papandreou, Liang-Chieh Chen, Kevin Murphy, and Alan L. Yuille. Weakly- and Semi-Supervised Learning of a DCNN for Semantic Image Segmentation. *arXiv:1502.02734 [cs]*, October 2015. arXiv: 1502.02734.

[10] Yinda Zhang, Shuran Song, Ping Tan, and Jianxiong Xiao. Panocontext: A whole-room 3d context model for panoramic scene understanding. pages 668–686, 2014.

[11] Ilija Mihajlovic. Everything you ever wanted to know about computer vision. Recuperado de: https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e, 2019.

[12] Piotr Mirowski, Matthew Koichi Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. Learning to navigate in cities without a map, 2018.

[13] Ebtsam Adel, Mohammed Elmogy, and Hazem El-Bakry. Image stitching based on feature extraction techniques: A survey. *International Journal of Computer Applications*, 99:1–8, 08 2014.

[14] Developer guide of Street View Static API. (s.f.). Recuperado de: https://developers.google.com/maps/documentation/streetview/intro.

[15] P. Guerra and P. Carballeira. Detección de objetos en imágenes urbanas de Google Street View(trabajo de fin de grado). *Escuela Politéctina Superior, Universidad Autónoma de Madrid*, 2019.

[16] P. Carballeira. Google Street View (GSV) database creator. Recuperado de: https://github.com/pcarballeira/gsv$database creator_fork$.

[17] Imagen segmentation guide. (s.f.). Recuperado de: https://www.fritz.ai/image-segmentation.

[18] Martin Thoma. A survey of semantic segmentation. 2016.

[19] Maneela Jain and Pushpendra Singh Tomar. Review of Image Classification Methods and Techniques. *International Journal of Engineering Research*, 2(8), 2013.

[20] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. Object Detection with Deep Learning: A Review. *CoRR*, April 2019.

[21] Sik-Ho Tsang. Review: DeepMask (Instance Segmentation). Recuperado de: https://towardsdatascience.com/review-deepmask-instance-segmentation-30327a072339, March 2019.

[22] Priya Dwivedi. Semantic segmentation - popular architectures. Recuperado de: https://www.fritz.ai/image-segmentation.

[23] Image Segmentation in Deep Learning: Methods and Applications. (s.f.). Recuperado de: https://missinglink.ai/guides/computer-vision/image-segmentation-deep-learning-methods-applications/.

[24] Verma G.K. Dhillon, A. Convolutional neural network: a review of models, methodologies and applications to object detection. *Prog Artif Intell*, 2019.

[25] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[26] The PASCAL Visual Object Classes Homepage. (s.f.). Recuperado de: http://host.robots.ox.ac.uk/pascal/VOC.

[27] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, Las Vegas, NV, USA, June 2016. IEEE.

[28] Gabriel J. Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV (1)*, pages 44–57, 2008.

[29] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[30] Javad Zolfaghari Bengar, Abel Gonzalez-Garcia, Gabriel Villalonga, Bogdan Raducanu, Hamed H Aghdam, Mikhail Mozerov, Antonio M Lopez, and Joost van de Weijer. Temporal coherence for active learning in videos. *arXiv preprint arXiv:1908.11757*, 2019.

[31] Yanming Guo, Yu Liu, Theodoros Georgiou, and Michael S. Lew. A review of semantic segmentation using deep neural networks. *International Journal of Multimedia Information Retrieval*, 7(2):87–93, June 2018.

[32] Manjot Kaur and Pratibha Goyal. A Review on Region Based Segmentation. 4(4):4, 2013.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, 06 2014.

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv:1506.01497 [cs]*, January 2016. arXiv: 1506.01497.

[35] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2017.

[36] Shu Liu, Xiaojuan Qi, Jianping Shi, Hong Zhang, and Jiaya Jia. Multi-scale Patch Aggregation (MPA) for Simultaneous Detection and Segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3141–3149, Las Vegas, NV, USA, June 2016. IEEE.

[37] Sik-Ho Tsang. Review: FCN — Fully Convolutional Network (Semantic Segmentation). Recuperado de: https://towardsdatascience.com/review-fcn-semantic-segmentation-eb8c9b50d2d1, March 2019.

[38] Wei Liu, Andrew Rabinovich, and Alexander C. Berg. Parsenet: Looking wider to see better, 2015.

[39] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network, 2016.

[40] Selim Seferbekov, Vladimir Iglovikov, Alexander Buslaev, and Alexey Shvets. Feature Pyramid Network for Multi-class Land Segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 272–2723, Salt Lake City, UT, USA, June 2018. IEEE.

[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[42] Deeplab: Deep Labelling for Semantic Image Segmentation. (s.f.). Recuperado de: https://github.com/tensorflow/models/tree/master/research/deeplab.

[43] Arthur Ouaknine. Review of Deep Learning Algorithms for Image (Semantic Segmentation). Recuperado de: https://medium.com/@arthur$_o$*uaknine*/$review - of - deep - learning - algorithms - for - image - semantic - segmentation - 509a600f7b57, 2018.

[44] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. *arXiv:1412.7062 [cs]*, June 2016. arXiv: 1412.7062.

[45] Sik-Ho Tsang. Review: DeepLabv1 DeepLabv2 — Atrous Convolution (Semantic Segmentation). Recuperado de: https://towardsdatascience.com/review-deeplabv1-deeplabv2-atrous-convolution-semantic-segmentation-b51c5fbde92d, November 2018.

[46] Sik-Ho Tsang. Review: DeepLabv3 — Atrous Convolution (Semantic Segmentation). Recuperado de: https://towardsdatascience.com/review-deeplabv3-atrous-convolution-semantic-segmentation-6d818bfd1d74, January 2019.

[47] Sik-Ho Tsang. Review: DeepLabv3+ — Atrous Convolution (Semantic Segmentation). Recuperado de: https://medium.com/@sh.tsang/review-deeplabv3-atrous-separable-convolution-semantic-segmentation-a625f6e83b90, September 2019.

[48] Deeplab with pytorch. (s.f.). Recuperado de: https://github.com/kazuto1011/deeplab-pytorch.

[49] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context, 2016.